# Le problème du mot

G.-G. Lucas, P. Hector, S. Béranger, V. Malachi, W. Lucas

8 janvier 2017

## 1 Définitions élémentaires

### 1.1 Mots

Soit $A$ un ensemble que nous nommerons *alphabet*. Les $n$-uplets d'éléments de $A$ seront nommés *mots* et l'ensemble des mots sera noté $A^*$. Le mot $(a_1, \ldots, a_n) \in A^n$ sera simplement noté $a_1 \ldots a_n$.

On a une opération naturelle de *concaténation*, représentée soit par un point, soit en juxtaposant simplement les mots, et définie par l'égalité :

$$(a_1 \ldots a_n) \cdot (b_1 \ldots b_n) = (a_1 \ldots a_n b_1 \ldots b_n)$$

Cette opération confère à $A^*$ une structure de monoïde, dont l'élément neutre est le mot vide noté $\epsilon$.

Dans toute la suite, un alphabet $A$ est fixé.

### 1.2 Systèmes de réécriture

Un système de réécriture $S$ est un ensemble de règles de la forme $a_i \to b_i$ (formellement, ce sont des couples, mais l'interprétation qu'on en fait justifie cette notation), où $a_i$ et $b_i$ sont deux mots.

On introduit une relation binaire sur les mots : soit $S = (a_i \to b_i)_{i \in I}$ un système de réécriture ; un mot $u$ peut être *réécrit* en un autre mot $v$, ce qu'on note aussi $u \to v$, lorsqu'il existe $u_1, u_2, v_1, v_2 \in A^*$ et $i \in I$ tels que $u = u_1 \cdot a_i \cdot u_2$ et $v = v_1 \cdot b_i \cdot v_2$.

Un mot $u$ tel qu'il n'existe aucun mot $v$ vérifiant $u \to v$ est dit *irréductible*.

La relation binaire $\to$ admet une clôture réflexive transitive $\to^*$ ; on peut aussi définir la relation symétrique $\leftrightarrow$ par $a \leftrightarrow b \Leftrightarrow (a \to b) \vee (b \to a)$ et considérer sa clôture réflexive transitive $\leftrightarrow^*$. Les interprétations de ces relations sont claires : par exemple, $u \to^* v$ signifie que, quitte à utiliser plusieurs règles de réécriture à la suite, on peut obtenir $v$ à partir de $u$. Nous dirons que deux mots $u$ et $v$ sont *équivalents* lorsque $u \leftrightarrow^* v$.

#### 1.2.1 Systèmes noethériens

Si la relation $\leftarrow$ est bien fondée, c'est-à-dire qu'il n'existe pas de chaîne infinie de mots $(u_i)_{i \in \mathbb{N}}$ telle que $\forall i, u_i \to u_{i+1}$, on dit que le système de réécriture est *noethérien*.

Si le système est noethérien, tout mot $u$ admet une forme irréductible, c'est-à-dire un mot $v$ irréductible tel que $u \to^* v$. Il suffit en effet de prendre un élément minimal (pour $\leftarrow$) de l'ensemble non vide $\{w \in A^* / u \to^* w\}$ puisque la relation est bien fondée. Toutes les formes irréductibles d'un mot donné sont naturellement équivalentes.

Étant donné un mot $u$ et un système de réécriture noethérien $S$, un algorithme simple permet d'obtenir une forme irréductible de $u$ : il suffit, tant que $u$ contient un $a_i$, de réécrire $u$ selon la règle $a_i \to b_i$. Cet algorithme

termine nécessairement car la relation est bien fondée, et le dernier mot obtenu ne contient aucun $a_i$ et est donc irréductible.

### 1.2.2 Systèmes confluents

Le système est dit *confluent* lorsque chaque fois qu'on a trois mots $u$, $v$ et $v'$ tels que $u \to^* v$ et $u \to^* v'$, il existe un quatrième mot $w$ tel que $v \to^* w$ et $v' \to^* w$. Autrement dit, si on réécrit un mot de deux façons différentes, il est toujours possible en réécrivant les deux résultats d'obtenir un même mot et donc de se rendre compte du fait que les mots sont équivalents sans avoir à « remonter » dans les règles de réécriture.

Si le système est confluent, un mot donné ne peut pas avoir plus d'une forme irréductible : en effet, si $v$ et $v'$ sont deux formes irréductibles d'un même mot, alors il existe un mot $w$ tel que $v \to^* w$ et $v' \to^* w$. L'irréductibilité donne alors $v = w$ et $v' = w$ d'où $v = v'$.

On a une forme plus faible de confluence, la *confluence locale*. Le système est localement confluent lorsque chaque fois qu'on a trois mots $u$, $v$ et $v'$ tels que $u \to v$ et $u \to v'$, alors il existe un quatrième mot $w$ tel que $v \to^* w$ et $v' \to^* w$. Cette propriété est nettement plus simple à vérifier, puisque les cas à traiter sont uniquement ceux où on applique une règle à la fois.

### 1.2.3 Systèmes noethériens et confluents

Si le système est noethérien et confluent, alors tout mot admet une unique forme irréductible, qu'on sait facilement déterminer. Deux mots équivalents ont des formes irréductibles équivalentes et donc égales[1], et on peut alors facilement déterminer si deux mots sont équivalents en vérifiant simplement si leurs formes irréductibles sont égales.

Un résultat, le lemme de Newman, nous assure en outre qu'un système noethérien localement confluent est confluent : en pratique, c'est cette propriété que nous utiliserons pour vérifier qu'un système est confluent, car la plupart des systèmes que nous allons considérer seront noethériens.

Notre stratégie pour étudier les systèmes de réécriture sera d'essayer de se ramener au cas des systèmes noethériens confluents chaque fois que ce sera possible.

### 1.2.4 Exemple

Avant d'aller plus loin, donnons un exemple simple. On considère l'alphabet $(a, b)$ et les règles (1) : $ab \to baa$ et (2) : $aab \to ba$. Alors : $abbaab \to_2 abbba \to_1 baabba \to_2 bbaba \to_1 bbbaaa$.

Sur cet exemple, on se rend compte qu'on a souvent plusieurs possibilités de réécriture, et qu'il faut donc « tâtonner » : on ne peut pas explorer l'arbre des possibilités de manière déterministe, et on s'attend donc à devoir parfois revenir en arrière.

On remarque aussi que la relation $abbaab \to^* bbaba$ n'est pas visible au coup d'œil : on a l'intuition que le problème algorithmique de l'équivalence de mots, bien que simple à formuler, soit non-trivial.

## 1.3 Présentation de groupes

### 1.3.1 Introduction, définition

La structure de monoïde libre obtenue sur $A^*$ est intéressante, et on comprend assez vite qu'en choisissant un système de réécriture adapté et en quotientant par la relation $\leftrightarrow^*$ (la concaténation étant bien compatible avec cette relation d'équivalence), on pourra représenter n'importe quel monoïde.

Un groupe étant un monoïde, on peut en particulier représenter les groupes. On commence par le groupe libre sur $A$ : en ajoutant dans l'alphabet $A$ des caractères de la formes $a^{-1}$ pour chaque $a \in A$, et en introduisant les règles $aa^{-1} \to \epsilon$ et $a^{-1}a \to \epsilon$, ledit quotient donne bien le groupe libre sur $A$.

1. En effet, on montre que deux mots irréductibles équivalents sont nécessairement égaux dans le cas d'un système confluent.

Si on a un groupe $G$ quelconque, engendré par une famille $(g_i)_{i\in I}$, alors en prenant pour $A$ l'ensemble des $g_i$ et des $g_i^{-1}$, c'est-à-dire qu'on ne considère les $g_i$ que comme des symboles formels, puis en prenant pour système de réécriture les règles $x \to \epsilon$ pour tous les mots $x = g_1^{\epsilon_1} g_2^{\epsilon_2} \dots g_n^{\epsilon_n} \in A^*$ ($\epsilon_i = \pm 1$) tels qu'on ait l'égalité $g_1^{\epsilon_1} g_2^{\epsilon_2} \dots g_n^{\epsilon_n} = 1_G$ dans $G$. Pour tous mots $u, v \in A^*$, on a $u \leftrightarrow^* v \Leftrightarrow u =_G v$, où $=_G$ désigne l'égalité des expressions évaluées dans $G$. Le quotient obtenu selon la méthode précédente sera donc bien un groupe isomorphe à $G$.

En fait, dans la plupart des cas, on n'a pas besoin d'autant de règles de réécriture. Par exemple, pour le groupe diédral $D_n$, engendré par $r$ et $s$, on peut se contenter des règles $r^n \to \epsilon$, $s^2 \to \epsilon$, $rsr \to s$. Une telle liste de générateurs et de règles caractérisant un groupe s'appelle une présentation du groupe.

Si on a un groupe $G$ et une présentation du groupe $G$ sous forme d'un alphabet $A$ et d'un ensemble de règles de réécriture de la forme $a_i \to b_i$, alors on peut obtenir un groupe isomorphe à $G$ en quotientant le groupe libre $L$ sur $A$, défini auparavant, par le sous-groupe distingué de $L$ engendré par les $a_i b_i^{-1}$ (cette définition n'utilise d'ailleurs pas explicitement la notion de réécriture). Une présentation contient donc toute l'information nécessaire pour qu'on puisse espérer comprendre le groupe à partir d'elle. Cela justifie donc, par exemple dans le cas du groupe diédral évoqué ci-dessus, de noter : $D_n =< r, s / r^n = s^2 = e, rsr = s >$.

### 1.3.2 Exemples, intérêt, enjeu

Cette façon de définir des groupes a pour principal intérêt de ne pas donner d'importance à la nature précise des éléments du groupe, mais simplement aux liens entre eux, qui suffisent à comprendre le groupe à isomorphisme près.

Un certain nombre de groupes sont définis par une présentation, et on peut alors les étudier formellement, sans avoir à donner un sens aux éléments du groupe : les groupes de Coxeter $C'_{r,m} =< s_1, \dots, s_r / \forall i \forall j s_i^2 = (s_i s_j)^m = e >$ ou les groupes d'Artin, par exemple, sont souvent étudiés sous cette forme.

De plus, cette façon de voir les groupes permet de les étudier à l'aide des système de réécriture, grâce à l'équivalence expliquée ci-dessus. Cela revient à manipuler des expressions sans passer au quotient, ce qui rend moins abstraite la notion d'égalité dans le groupe (incarnée par l'équivalence des mots) et permet d'employer les notions de confluence et de noethérianité.

La principale difficulté de cette vision, qu'on a déjà évoquée, est qu'il n'est même pas toujours aisé de déterminer si, sous forme d'expressions, deux mots correspondent au même élément du groupe (c'est-à-dire qu'ils sont équivalents). C'est cette difficulté qui donne lieu au problème du mot.

## 2 Problème du mot

Le problème du mot est la recherche d'une réponse à la question de l'équivalence entre deux mots. Il a été largement étudié et ceci sous différents aspects : algébriquement, informatiquement, et parfois géométriquement (via les graphes de Cayley). Si on a parfois des solutions partielles, dans le cas de groupes particuliers, et qu'on peut s'intéresser spécifiquement à des cas comme les groupes de Coxeter ou d'Artin, nous n'avons pas de solution au problème en général, et pour cause : celui-ci est indécidable dans le cas des groupes (et des monoïdes). Ce résultat est dû à Novikov, qui l'a publié en 1955.

Cette indécidabilité signifie qu'il n'existe pas d'algorithme donnant dans tous les cas une réponse à la question « avec ces règles de réécriture, ces deux mots sont-ils équivalents ? ».

Essayons de réfléchir à ce à quoi pourrait ressembler une solution naïve au problème : on explorerait, par un parcours en largeur (i.e. en explorant parallèlement toutes les possibilités), toutes les branches de l'arbre qu'on obtient en partant d'un mot $u$ et en voyant comme « fils » d'un mot $a$ tous les mots $b$ tels que $a \leftrightarrow b$, puis on chercherait à déterminer si $v$ est présent dans cet arbre. Ce parcours en largeur est effectivement programmable, et en cas d'équivalence on obtiendrait bien une réponse en temps fini, il existe néanmoins en général des branches arbitrairement grandes dans l'arbre (même lorsqu'il n'y a pas de branches infinies [2]) et le procédé ne s'arrêterait donc jamais lorsque la réponse est non : cette méthode naïve ne définit donc pas un algorithme. Déterminer si

2. Si en revanche on se pose la question de savoir $a \to^* b$, lorsqu'il y a un nombre fini de règles et que le système est noethérien, l'arbre obtenu est à branchement fini et sans branche infinie, donc la contraposée du lemme de König nous assure qu'il est fini : on peut donc répondre.

cette procédure s'arrête, c'est exactement répondre au problème du mot : l'indécidabilité du problème de l'arrêt nous donne donc une idée des raisons profondes qui rendent ce problème indécidable (même si cela ne prouve rien, car un algorithme plus sophistiqué pourrait malgré tout convenir).

Dans les cas particuliers simples (groupe libre, abélien libre, diédral), on a néanmoins une forme réduite simple de chaque élément qu'on peut comparer pour répondre au problème du mot. Pour le groupe diédral $D_n$, on a par exemple la forme $s^\alpha r^\beta$ avec $\alpha \in \{0,1\}$ et $r \in \{0,\dots,n-1\}$.

Profitons-en pour signaler que le problème du mot n'intervient pas qu'en algèbre : en logique, il peut aussi servir pour formaliser certains types de preuves. On peut même voir certaines règles du calcul intégral (l'intégration par parties, le changement de variables, ...) comme des règles de réécriture, et le problème du mot se pose donc même en analyse. Les questions que ce problème engendre sont de tous types et peuvent se trouver au cœur des mathématiques (voir à ce sujet la section 6).

### 2.1 Remarque sur l'orientation des règles

Le problème du mot pose une question sur la relation $\leftrightarrow^*$, qui ne dépend pas de l'orientation qu'on donne aux règles (puisqu'on interprète celles-ci comme des égalités). L'orientation des règles n'est donc pas à voir comme une donnée du problème, mais comme quelque chose qu'on peut modifier lorsqu'on veut avoir de meilleurs propriétés (comme la noethérianité, la confluence, l'irréductibilité, etc. qui sont des notions qui dépendent de l'orientation).

## 3 Cas du groupe de tresses

*Pour la définition du groupe de tresses et ses propriétés, voir l'exposé correspondant. Nous essaierons néanmoins de faire des dessins pour illustrer ce qu'on fait. Nous suivons le plan du film Braids, disponible gratuitement sur le site de l'université de Trente :* `http://matematita.science.unitn.it/braids/summary.html`

*Dans une tresse, nous numéroterons les brins de* 1 *à* $n$ *en partant du haut.*

Le principal intérêt du cas du groupe de tresses est qu'on peut voir géométriquement ce qu'on fait pour réduire nos mots, c'est donc moins abstrait et plus palpable. C'est un cas particulier d'une grande importance historique, car il s'agit d'un des cas non-triviaux où l'on sait résoudre le problème du mot.

On peut présenter le groupe de tresses $B_n$ comme $< \sigma_1,\dots,\sigma_{n-1} / \sigma_i\sigma_{i+1}\sigma_i = \sigma_{i+1}\sigma_i\sigma_{i+1}, |i-j|>1 \Rightarrow \sigma_i\sigma_j = \sigma_j\sigma_i >$ ($\sigma_i$ représentant « l'inversion » entre les brins aux positions $i$ et $i+1$, le brin $i$ passant devant, et l'égalité étant à isotopie près).

Si la présentation est une étape obligatoire dans l'étude de ce groupe, nous allons néanmoins tenter de résoudre le problème du mot en ne pensant pas aux $\sigma_i$, mais aux tresses elles-mêmes, afin de développer une intuition géométrique du problème.

Il se trouve que, sans être trivial, le problème du mot est décidable dans le cas du groupe de tresses. Nous allons présenter deux algorithmes, informellement et sans preuve, permettant de décider si une tresse quelconque est isotope à la tresse triviale (pour savoir si $u \leftrightarrow^* v$, il suffit de vérifier que $uv^{-1} \leftrightarrow^* \epsilon$)

### 3.1 Premier algorithme : le peignage d'Artin

Nous appelons *pure* une tresse qui préserve l'ordre des brins.

Si une tresse n'est pas pure, ce qui est simple à vérifier, on sait immédiatement qu'elle n'est pas équivalente à la tresse triviale, et on n'a donc qu'à considérer l'autre cas.

On peut montrer que toute tresse pure de $B_n$ peut être écrite comme un produit $a_1 \dots a_{n-1}$ où, dans la tresse pure $a_i$, tous les brins ne bougent pas, sauf le brin $i$ qui peut s'emmêler avec tous les brins qui se trouvent en-dessous de lui (et pas uniquement celui immédiatement en-dessous). Une telle décomposition s'appelle forme normale. Trouver une forme normale va nous permettre de résoudre le problème.

Pour obtenir une forme normale, nous utiliserons une procédure (qui prouve au passage l'existence de la forme normale) due à Artin (1930) : le « peignage de tresses » :

Soit une tresse $u$. On commence par supprimer complètement le premier brin de la tresse en le remplaçant par un brin parfaitement horizontal, on obtient alors une tresse $u'$. Celle-ci vérifie $u'^{-1}u' \leftrightarrow^* \epsilon$, donc $uu'^{-1}u' \leftrightarrow^* u$. En simplifiant $uu'^{-1}$, on obtient un bloc dont tous les brins sont droits sauf le premier : c'est bien la forme voulue. On peut même faire en sorte que le premier brin entoure successivement les autres brins en revenant à sa position initiale entre chacun : en posant $g_i = \sigma_1 \dots \sigma_{i-1}\sigma_i\sigma_i\sigma_{i-1}^{-1} \dots \sigma_1^{-1}$, on peut écrire $uu'^{-1}$ comme un produit de $g_i$. En recommençant cette étape avec $u'$ et le second brin (et en ignorant le premier brin qui est horizontal), et ainsi de suite, on obtient alors bien la forme normale souhaitée.

L'avantage de cette forme normale est que la tresse triviale n'admet pas d'autre forme normale qu'elle-même : en mettant une tresse sous forme normale, il est alors immédiat de voir si elle est équivalente ou non à la tresse triviale : il suffit de voir si elle y est égale (on peut en fait s'arrêter dès qu'on trouve un bloc différent de la tresse triviale) !

Cet algorithme souffre cependant d'un gros problème : sa complexité est exponentielle et les temps obtenus vraiment élevés, ce qui ne laisse pas croire en une application pratique concrète.

### 3.2 Deuxième algorithme : la réduction des poignées

En 1995, P. Dehornoy propose un autre algorithme : l'algorithme de la réduction des poignées. Celui-ci a aussi une complexité théorique exponentielle dans le pire des cas, mais est le plus rapide dont on dispose.

Il s'agit d'identifier des *poignées* dans les tresses, c'est-à-dire des facteurs de la forme $\sigma_i^\epsilon x \sigma_i^{-\epsilon}$ avec $\epsilon \in \{-1, 1\}$ et $x \in < \sigma_1, \dots \sigma_{i-1} >$. En gardant ces notations, on applique alors à la tresse le morphisme de groupes $\phi$ qui à $\sigma_i$ associe la tresse triviale, qui à $\sigma_{i-1}$ associe $\sigma_{i-1}^{-\epsilon}\sigma_i\sigma_{i-1}^{\epsilon}$ et qui garde les autres $\sigma_j$ intacts (cela le définit entièrement). Graphiquement, il s'agit en fait de « retourner la poignée ». Tant qu'il existe une poignée, on la retourne avec ce morphisme, ce qui permet de l'éliminer.

Dehornoy a prouvé que l'image d'une tresse par ce morphisme est équivalente à la tresse initiale, qu'une tresse non vide sans poignée est nécessairement non triviale, et que l'opération décrite ci-dessus termine.

Ainsi, on élimine les poignées une à une, et la forme finale obtenue permet de conclure : la tresse avec laquelle on est parti est triviale si et seulement si la tresse obtenue à la fin est vide.

On trouve des détails supplémentaires sur ces algorithmes (et, ce qui manque le plus ici, des illustrations) dans les pages 15 à 19 de ce document : `http://www.math.toronto.edu/dalvit/pdf/PopularizationBraids.pdf`.

## 4 Procédure de Knuth-Bendix

### 4.1 Fonctionnement

Bien que le problème du mot soit insoluble dans le cas général, il existe des algorithmes [3] qui apportent une solution partielle au problème. Les algorithmes de Todd-Coxeter et de Knuth-Bendix en sont des exemples. Nous allons présenter brièvement ce dernier dans le cas des monoïdes de type fini. Une des qualités de celui-ci est que son principe de fonctionnement nécessite que l'utilisateur ordonne les mots, ce qui le rend potentiellement plus performant puisqu'on peut choisir l'ordre de manière intelligente.

Le principe est, en partant d'un système de réécriture $S$ fini, sur un alphabet $A$ fini lui-aussi, de trouver un système confluent équivalent (c'est-à-dire qu'il donne lieu à la même relation $\leftrightarrow^*$). Pour cela, l'utilisateur doit munir $A^*$ d'un ordre total bien fondé compatible avec la concaténation [4], et vérifiant (quitte à changer le sens des règles) $a_i > b_i$ pour chaque règle $a_i \to b_i$ (le système obtenu est alors noethérien).

D'abord, on fait en sorte que les $b_i$ et les $a_i$ soient tous irréductibles, en appliquant des réductions jusqu'à ce que ce soit le cas, et en réorientant à nouveau les règles s'il le faut.

3. Il ne s'agit en fait pas d'algorithmes, car on ne peut pas assurer leur terminaison. Il semble néanmoins que cet abus de langage soit la norme.

4. C'est-à-dire que si on a $u < v$, alors pour tous mots $x$ et $y$, on a $xuy < xvy$.

Nous dirons que deux règles $a_i \to b_i$ et $a'_i \to b'_i$ $(i \neq j)$ se superposent lorsque :
— Il existe $u, v, w \in A^*$ avec $v \neq \epsilon$ tels que ($a_i = v$ et $a_j = uvw$) ou ($a_i = vw$ et $a_j = uv$) ou ($a_j = v$ et $a_i = uvw$) ou ($a_j = vw$ et $a_i = uv$)
— En réécrivant le mot $uvw$ d'une part avec la règle $a_i \to b_i$ et d'autre part avec la règle $a'_i \to b'_i$, puis en calculant une forme irréductible de chacun des mots obtenus, on obtient deux mots $r_1$ et $r_2$ différents[5].

**Théorème 1.** *Le système est confluent si et seulement s'il n'existe pas de règles de $S$ qui se superposent (on dit aussi que toutes les paires critiques sont joignables).*

**Preuve 1.** *L'implication $\Rightarrow$ est évidente. Vérifions que la propriété de droite implique la confluence locale : soit un mot a qu'on peut réécrire avec deux règles. Si les règles s'appliquent à des sous-mots disjoints, il suffit pour avoir la confluence locale d'en utiliser l'une puis l'autre et vice-versa. Sinon, les règles s'appliquent sur un même sous-mot, elles vérifient donc la première partie de la définition de règles se superposant et a contient le uvw correspondant, on peut donc ne vérifier la confluence locale que pour les uvw : on obtient celle-ci directement puisque les formes irréductibles des deux façons de réécrire uvw sont égales (les règles ne se superposent pas).*

Il faut comprendre les règles qui se superposent comme des règles pouvant s'appliquer à un même mot en causant des non-confluences. Notre objectif est donc d'éliminer les superpositions de règles.

Nous avons toutes les clés en main pour détailler l'algorithme de Knuth-Bendix, qui prend en entrée un système de réécriture $S_0$.

1. On pose $S = S_0$
2. On répète les opérations suivantes :
   (a) On pose $S' = S$
   (b) Pour chaque paire de règles se superposant dans $S'$ (et pour chaque manière de les superposer), en reprenant les notations de la définition de la superposition, on ajoute à $T$ la règle $max(r_1, r_2) \to min(r_1, r_2)$[6] puis on rend irréductibles les $a_i$ et les $b_i$ devenus réductibles, en réorientant les règles s'il le faut.

   et ce jusqu'à ce que $S = S'$ (c'est-à-dire que $S$ n'évolue plus, et qu'on a donc atteint un état stationnaire)

À chaque étape, la superposition considérée n'existe plus, et le théorème énoncé nous donne l'impression que ce processus nous rapproche donc bien d'une solution. Malgré cette intuition, le procédé ne termine pas toujours. En particulier, le fait de rajouter des règles à chaque fois est susceptible d'ajouter des paires de règles se superposant, et cela peut très bien ne jamais finir. **Si l'algorithme termine**, vu le théorème, le système obtenu est confluent, noethérien et équivalent au système initial, ce qui permet de répondre au problème du mot en comparant les formes irréductibles.

## 4.2 Exemple

Considérons l'alphabet $\{a, b, c\}$, avec l'ordre lexicographique, et les règles (1) : $bab \to aba$ et (2) : $ab \to c$. Après simplification et réorientation, on a : (1) : $ca \to bc$ et (2) : $ab \to c$.

Ces deux règles se superposent car $cab \to_1 bcb$ et $cab \to_2 cc$. On ajoute donc la règle (3) : $bcb \to cc$.

(2) et (3) se superposent car $abcb \to_2 ccb$ et $abcb \to_3 acc$. On ajoute donc la règle (4) : $ccb \to acc$.

(3) et (3) ne se superposent pas car on a $(bcb)cb \to_3 cccb \to_4 cacc \to_1 bccc$ et $bc(bcb) \to_3 bccc$.

(3) et (4) ne se superposent pas car $ccbcb \to_4 acccb \to_4 acacc \to_1 abccc \to_2 cccc$ et $ccbcb \to_2 cccc$.

On n'a plus aucune superposition. Le système obtenu est donc confluent. On peut faire des essais : a-t-on par exemple $bcbacbc$ et $cbcbabba$ équivalents ? On peut répondre de manière mécanique :

$$bcbacbc \to_3 ccacbc \to_1 cbccbc \to_4 cbaccc$$

$$cbcbabba \to_3 cccabba \to_2 ccccba \to_4 ccacca \to_1 cbccca \to_1 cbccbc \to_4 cbaccc$$

5. La forme irréductible obtenue n'a pas trop d'importance, même si cela fait de la superposition des règles une notion mal définie puisqu'elle dépend des choix faits en réduisant.
6. Cela change la relation $\to^*$, mais pas $\leftrightarrow^*$.

Les deux mots sont donc équivalents. Trouver un chemin entre eux dans le système initial n'est pourtant pas de tout repos !

# 5 Lien avec l'algorithme de Buchberger

*Une bonne partie du contenu de cette section est issue du compte-rendu de TIPE suivant : `http://math.univ-lyon1.fr/~caldero/Robin.pdf`, qu'on pourra consulter pour avoir davantage de preuves.*

## 5.1 Présentation de l'algorithme

L'algorithme de Buchberger est destiné à calculer des bases de Gröbner d'ideaux d'un anneau de polynômes.

Plaçons-nous dans l'anneau de polynômes $K[x_1, \ldots, x_n]$ d'un corps $K$. Nous munissons l'ensemble $\mathcal{M}$ des monômes de cet anneau d'un ordre total adapté, c'est-à-dire qu'il vérifie : $\forall M, N \in \mathcal{M}, (M \leq N \Rightarrow \forall P \in \mathcal{M}, MP \leq NP)$ et $\forall M, N \in \mathcal{M}, M \leq MN$. Un tel ordre est nécessairement bien fondé. On peut utiliser l'ordre lexicographique, mais d'autres choix peuvent être plus adaptés.

Étant donné un polynôme $P \in K[x_1, \ldots, x_n]$, il devient possible de parler du *terme de tête* (maximal pour l'ordre monomial), qu'on note $LT(P)$. Le monôme correspondant (unitaire) est noté $LM(P)$. Par exemple, $LT(3xy - x) = 3xy$ et $LM(3xy - x) = xy$.

Si on se donne les règles de réécritures[7] $LT(f) \to LT(f) - f$ pour chaque polynôme $f$ d'une base $B$ d'un idéal $I$, le système est noethérien[8] et les réécritures ne changent pas l'appartenance à $I$. Lorsque ce système est confluent, la base $B$ est dite de Gröbner. Trouver une base de Gröbner d'un idéal a donc un lien naturel avec le fait de trouver un système noethérien confluent équivalent à celui obtenu à partir d'une base quelconque de l'idéal (qui est déjà noethérien) : le problème semble proche du précédent.

Quand $B$ est une base de Gröbner de $I$, on remarque que le système de réécriture $\{LT(f) \to LT(f) - f / f \in B\}$ donne comme réduction la division euclidienne par l'idéal $I$, car chaque réécriture est une étape de l'algorithme de division euclidienne (on peut s'en convaincre en regardant le cas $n = 1$, $|B| = 1$).

On peut aussi définir une base de Gröbner de $I$ comme une partie $G \subset I$ vérifiant $< \{LT(f)/f \in G\} >=< \{LT(f)/f \in I\} >$. L'inclusion $\subset$ est vérifiée dès que $G$ est une base de $I$, tandis que l'inclusion réciproque revient à dire que $G$ a de bonnes propriétés qui font que ses éléments sont assez "représentatifs" de tous ceux de $I$ (par exemple, ils ne peuvent pas avoir un terme de tête commun sans que celui-ci soit aussi le terme de tête de tous les éléments de $I$), ce qui permet de les réduire tous suffisamment pour qu'on n'ait pas plusieurs formes irréductibles. Nous admettons l'équivalence des deux définitions sans preuve, ainsi que le fait que tout idéal admette une base de Gröbner (qui peut être vu comme une conséquence de l'algorithme suivant).

Étant donné deux polynômes $f$ et $g$, on pose $D(f, g) = LM(f) \vee LM(g)$ puis $S(f, g) = f\frac{D(f,g)}{LT(f)} - g\frac{D(f,g)}{LT(g)}$, où $\vee$ désigne le PPCM unitaire des monômes (la puissance de chaque $x_i$ est le maximum des deux puissances).

**Théorème 2** (Critère de Buchberger). *Une base $F = (f_1, \ldots, f_t)$ d'un idéal $I$ est une base de Gröbner si et seulement si, en reprenant la règle de réécriture définie ci-dessus, on a $\forall i \forall j, S(f_i, f_j) \to^* 0$.*

**Preuve 2.** *Remarquons déjà qu'on a les réécritures suivantes :*

$$D(f_i, f_j) = uLT(f_j) \to u(LT(f_j) - f_j) = D(f_i, f_j) - uf_j$$

$$D(f_i, f_j) = vLT(f_i) \to v(LT(f_i) - f_i) = D(f_i, f_j) - vf_i$$

*Si le système est confluent, $uf_j$ et $vf_i$ ont donc une même forme irréductible et leur différence $S(f_i, f_j)$ peut ainsi bien être réécrite en 0. Vérifions la réciproque en montrant la confluence locale : si un polynôme peut être réécrit par deux règles (associées aux polynômes $f_i$ et $f_j$) sans qu'elles n'agissent que sur des termes disjoints, c'est qu'elles agissent sur un terme multiple à la fois de $LT(f_i)$ et de $LT(f_j)$, c'est-à-dire multiple de $D(f_i, f_j)$.*

7. La règle $A \to B$ signifie « $P \to Q$ lorsqu'il existe $U$ et $V$ deux polynômes tels que $P = AU + V$ et $Q = BU + V$ ». Essentiellement, cela revient à dire qu'on peut remplacer toute occurrence de $A$ dans l'écriture d'un polynôme par un $B$.

8. Lorsqu'on a $P \to Q$ dans ce système, un des termes est réécrit en un polynôme dont le terme de tête est strictement plus petit, car les termes de tête de $LT(f)$ et $f$ se compensent dans $LT(f) - f$. Le fait que l'ordre monomial soit bien fondé assure donc la noethérianité.

*Il suffit donc de montrer la confluence locale pour les $D(f_i, f_j)$, ce qui vu le calcul précédent est une conséquence de la relation $S(f_i, f_j) \to^* 0$.*

Ce théorème est parfaitement analogue au théorème prouvé dans la présentation de l'algorithme de KNUTH-BENDIX (jusque dans sa preuve). Les paires $(f_i, f_j)$ telles que $S(f_i, f_j) \twoheadrightarrow^* 0$ jouent donc un rôle exactement similaire à celui des règles se superposant, et on cherche donc à les éliminer. En adaptant légèrement l'algorithme de KNUTH-BENDIX, on obtient alors l'algorithme de BUCHBERGER, qui prend un idéal $I$ de $K[x_1, \ldots, x_n]$ quelconque et une base $F$ de cet idéal, et renvoie une base de GRÖBNER de cet idéal. Voilà son fonctionnement :

1. On pose $G = F$.
2. On répète les opérations suivantes :
   (a) On pose $G' = G$
   (b) Pour chaque paire $\{p, q\} \subset G$, si la forme irréductible $S'$ obtenue en réduisant $S(p, q)$ avec le système de réécriture associé à $G'$ est non nulle (on dit que la paire est critique), alors on ajoute $S'$ à $G$ [9].

   et ce jusqu'à ce que $G = G'$

Cet algorithme termine toujours [10], et la famille $G = (g_1, \ldots, g_t)$ obtenue à la fin vérifie bien que pour tous $i$ et $j$ on a $S(g_i, g_j) \to^* 0$ : c'est donc une base de GRÖBNER.

Nous ne cherchons pas des bases de GRÖBNER pour le plaisir : celles-ci ont un très grand intérêt dans les problèmes d'algèbre, et en particulier pour répondre à la question de savoir si un polynôme donné appartient ou non à l'idéal engendré par une certaine base : en effet, une fois qu'on a une base de GRÖBNER, cela correspond simplement à vérifier que 0 est la forme irréductible du polynôme, ce qu'on sait faire simplement puisque le système est noethérien et confluent. Grâce à ces bases, on peut donc répondre à la question de l'appartenance à un idéal de manière algorithmique.

Nous avons mis en valeur autant que possible le lien profond entre ce problème et le problème du mot, on reste miraculeux que l'algorithme de BUCHBERGER, qui est une traduction presque littérale de celui de KNUTH-BENDIX, termine dans tous les cas, alors qu'on aurait pu craindre qu'il souffre des mêmes défauts que ce dernier.

### 5.2 Remarques sur le lien entre les deux algorithmes

Nous en avons déjà dit beaucoup sur ce qui rassemble les deux algorithmes ci-dessus, et on pourrait encore en dire beaucoup. Si on regarde attentivement ce qu'on calcule à chaque étape dans les deux algorithmes, on remarque que ce sont en fait *exactement* les mêmes opérations que nous faisons : l'équivalence entre les algorithmes n'est donc pas purement théorique, mais se reflète concrètement.

On peut encore développer ce lien. Par exemple, si on cherche des bases de GRÖBNER *minimales*, on se rend compte que cela équivaut à chercher des systèmes de réécriture *réduits*. Cela permet de répondre de manière parfaitement parallèle aux deux questions « Ces deux bases engendrent-elles le même idéal ? » et « Ces deux systèmes de réécriture sont-ils équivalents ? ». On peut donc pousser l'analogie entre les bases d'idéaux d'un anneau de polynôme et les systèmes de réécriture encore plus loin.

## 6 Pour aller plus loin

Sans rentrer dans les détails, signalons plusieurs choses intéressantes sur les systèmes de réécriture et le problème du mot :

— Les groupes définis par des présentations, qui peuvent être très intéressants algébriquement, ne peuvent être bien compris qu'en comprenant assez bien les systèmes de réécriture pour avoir une idée de la structure de ces groupes. C'est en particulier important de faire cette démarche dans le cas des groupes de COXETER et d'ARTIN. On peut regarder à ce sujet la présentation suivante : `https://lmb.univ-fcomte.fr/IMG/pdf/expose_olivier-3.pdf`.

9. Cela ne change pas l'idéal, puisque $S'$ y appartient déjà (il est équivalent à 0).
10. Les raisons sont complexes, mais essentiellement, cela provient du fait que l'idéal $< \{LT(f)/f \in G\} >$ grossit, et qu'il doit donc stationner car l'anneau $K[x_1, \ldots, x_n]$ est noethérien.

— Comme nous l'avons rapidement dit, il est possible de formaliser certains types de preuves par des systèmes de réécriture (les axiomes sont alors des règles). Cela permet au moins dans une certaine mesure de comprendre l'indécidabilité du problème du mot comme un analogue du théorème d'incomplétude de Gödel. Les preuves assistées par ordinateur et la vérification informatique de preuves sont des domaines de recherche actifs, et ce point est donc assez important. On peut se référer à : `http://comjnl.oxfordjournals.org/content/34/1/2.full.pdf`
— Il est possible d'avoir une compréhension plus large des problèmes dont nous parlons en les inscrivant dans le cadre de la théorie des catégories et des bases d'homotopies ; en particulier, on peut utiliser un résultat assez abstrait nommé « théorème de Squier », qui donne une borne sur le rang du troisième groupe d'homologie[11] d'un monoïde, pour compléter et réduire des systèmes de réécriture. On peut consulter : `http://thaega.progval.net/rapport.pdf` (qui implémente Knuth-Bendix en OCaml !) et `http://iml.univ-mrs.fr/~lafont/pub/agr.pdf`.
— Le nombre minimal de réécritures nécessaires pour lier deux mots équivalents, qui est une distance, peut être un objet d'étude de la théorie géométrique des groupes, qui permet de donner des informations de nature topologique, par exemple sur le graphe de Cayley.
— On pourra aussi lire une preuve « simplifiée » de l'indécidabilité du problème du mot ici : `http://www.personal.psu.edu/t20/logic/seminar/050517.pdf`

La liste n'est bien sûr pas exhaustive et le point important est de comprendre que les systèmes de réécriture et le problème du mot sont au cœur de bien des questions mathématiques, comme en témoigne le fait qu'on puisse tisser des liens multiples entre eux et la plupart des domaines de l'algèbre.

Les bases de Gröbner ont elles-aussi des applications étonnantes. On peut par exemple s'en servir pour résoudre des Sudoku. Voir à ce sujet le document suivant :
`https://pdfs.semanticscholar.org/45bc/787bb68ee95849fd5f2b8332fbd1a53ead67.pdf`.

# 7 Sources

Depuis Wikipedia :
— (fr) Problème du mot
— (fr) Complétion de Knuth-Bendix
— (fr) Paire critique
— (fr) Base de Gröbner
— (en) Abstract rewriting system
— (en) Word problem
— (en) Word problem for groups
— (en) Braid group
— (en) Knuth-Bendix completion algorithm
— (en) Gröbner basis

`https://www.youtube.com/watch?v=pHug5Gb7FaA`
`http://www.math.toronto.edu/dalvit/pdf/PopularizationBraids.pdf`
`http://math.univ-lyon1.fr/~caldero/Robin.pdf` (lien avec Buchberger)
`https://pdfs.semanticscholar.org/fb56/b9e4898870d6f1ff5414cc2891e678e49c8b.pdf`

11. Décidément, nous allons parvenir à établir des liens avec tous les autres exposés !